

# Python, I deo

© Predrag Pejović, 

# Python?

- ▶ programski jezik
- ▶ Wikipedia:
  - ▶ “Python is a general-purpose, high-level programming language whose design philosophy emphasizes code readability. Python claims to “[combine] remarkable power with very clear syntax”, and its standard library is large and comprehensive. Its use of indentation for block delimiters is unique among popular programming languages.”
  - ▶ “The reference implementation of Python (CPython) is free and open source software and has a community-based development model, as do all or nearly all of its alternative implementations. CPython is managed by the non-profit Python Software Foundation.”

# Python??

- ▶ interpreter, scripting language
- ▶ po tome nalik na BASIC (nekada), Octave, ...
- ▶ nema kompilacije i linkovanja, vrlo brze probe
- ▶ sporije od C-a
- ▶ ali se dobro povezuje sa C-om
- ▶ jako moćne i raznovrsne biblioteke (pySerial, numpy, matplotlib, sympy, ...)
- ▶ jednostavna sintaksa
- ▶ opšta namena
- ▶ free!!!
- ▶ jako dobro podržan, razvija se, rasprostranjen
- ▶ Google, Youtube, ...
- ▶ svaka distribucija GNU/Linux-a ga ima

# Python???

- ▶ Guido van Rossum, December 1989
- ▶ masovno se uči kao prvi programski jezik: MIT, CU Boulder, ...
- ▶ radi pod raznovrsnim platformama, sve koje se kod nas sreću obuhvaćene
- ▶ vrlo objektno orijentisan, mada ne mora da se koristi
- ▶ vrlo moćni tipovi podataka
- ▶ lako se prave novi tipovi podataka

# Python, kako nabaviti? GNU/Linux

GNU/Linux:

- ▶ već ima interpreter, sigurno
- ▶ provera: komandna linija, `python3` nekad bilo `python`  
`Python 3.8.5 (default, Jul 28 2020, 12:59:40)`  
`[GCC 9.3.0] on linux`  
`Type "help", "copyright", "credits" or "license" for`  
`more information.`
- ▶ nešto valja dovući iz repository (za Python 3!):
  - ▶ IDLE
  - ▶ IPython
  - ▶ numpy
  - ▶ scipy
  - ▶ matplotlib
  - ▶ pylab (sve prethodno)
  - ▶ python-serial
  - ▶ Sympy
  - ▶ Spyder
  - ▶ ...

# Python, kako nabaviti? win

Windows:

- ▶ <http://python.org/>
- ▶ dovcete, instalirate
- ▶ za win je IDLE included
- ▶ ostalo?
  - ▶ ili <https://pypi.org/project/pip/>
  - ▶ ili <https://www.anaconda.com/products/individual>
  - ▶ ipython+numpy+scipy+matplotlib+R+...
  - ▶ pySerial, SourceForge,  
<https://pypi.python.org/pypi/pyserial>
  - ▶ SymPy, <http://sympy.org/en/index.html>
  - ▶ Spyder, <https://pypi.python.org/pypi/spyder>
  - ▶ ...

# Python, 2 ili 3? 3!

- ▶ rešeno, verzija 2 više „nije podržana“
- ▶ forking, 3 je „nov“ jezik
- ▶ verzija 3 nema backward compatibility
- ▶ nisu prevelike razlike (print, input, range, za početak)
- ▶ problem sa već napisanim programima
- ▶ problem ako se oslanjate na već postojeće programe
- ▶ koristim numpy, matplotlib, ... pylab
- ▶ bilo neočekivano brzo prebačeno
- ▶ verzija 3 je aktuelna, nju predajem
- ▶ **python3**

Python 3.8.5 (default, Jul 28 2020, 12:59:40)

[GCC 9.3.0] on linux

Type "help", "copyright", "credits" or "license" for more information.

# Python, dokumentacija

- ▶ <http://www.python.org/doc/>, sve što treba
- ▶ <http://ocw.mit.edu/>, Python kursevi, ima više
- ▶ isto, edX
- ▶ <https://openbookproject.net/thinkcs/python/english3e/>
- ▶ <https://greenteapress.com/wp/think-python-2e/>
- ▶ još mnogo free resursa, realno je samo #1 potrebno
- ▶ izbor ostalog izlistan na sajtu predmeta



# Python, dokumentacija, realno

Ako ne učite programiranje, već programski jezik:

- ▶ <http://www.python.org/doc/>
- ▶ A4, pdf, .zip, 15.2 MB
- ▶ Python 3.9.1rc1, December 03, 2020
- ▶ tutorial.pdf, Python Tutorial, 147 strana
- ▶ reference.pdf, The Python Language Reference, 168 strana
- ▶ library.pdf, The Python Library Reference, 2084 strane
- ▶ ↑ ovde je suština uspeha

# valja pomenuti i ...

## 1. PyCharm, <https://www.jetbrains.com/pycharm/>

- ▶ popularno ...
- ▶ Editions:
  - 1.1 Professional (proprietary!)
  - 1.2 Community ("Open Source")
- ▶ ne koristim ga, nekompetentan za komentare

## 2. Thonny, <https://thonny.org/>

- ▶ ima ga Raspbian, RaspberryPi
- ▶ ne koristim, samo pogledao, ...

# Python, počinjemo, kalkulator

Pokrenete IDLE ili ipython, kako god znate (kom. lin., dash, ...)

osnovne operacije:

2+2

2-3

2\*3

od verzije 3 nema iznenađenja:

3/4\*100

3.0/4.0\*100.0

3.0/4\*100

3./4\*100

# Python, mislili ste da je sa deljenjem gotovo?

10//3

10.0/3.0

10.0//3.0

-10.0//3.0

-10//3

# Python, type i celobrojno deljenje

```
help(type)
type
type(3)
type(3.0)
type(3.)
type(10//3)
type(10.0//3)
type(10/3.)
type(10./3.)
```

# Python, stepenovanje

`2^3`

`3^2`

`3^3`

`10^10`

`2**3`

`2 ** 3`

`3 ** 2`

`10 ** 10`

`type(10**10)`

`3**64`

`type(3**3)`

`type(3**64)`

# Python, ostatak pri deljenju

```
10%3
11%3
12%3
t=54+12
print(t)
type(t)
s=t//60
m=t%60
print(s)
print(m)
print(s, m)
print('proteklo je', s, 'sat i', m, 'minuta')
```

# Python, operatori poredjenja

2 == 2

2==2

3 == 2

2 != 3

2 != 2

#2 <> 2

#2 <> 3

2 > 3

2 < 3

2 >= 1

2 >= 2

2 >= 3

2 <= 1

2 <= 2

2 <= 3



# Python, logičke operacije, :, \ i #

```
type(True); type(False)
a = True
b = False
type(a)
a and b    # logicko i
not a      # logicko ne
a and a
a or not a
a or (not a)
a or \
not b      # ovako se nastavlja red
```

# Python, zapisi brojeva

#012

0o12

0012

0x35

0X35

0b11

0B11

# Python, konverzija zapisa brojeva

```
oct(10)
```

```
hex(53)
```

```
bin(3)
```

# Python, da račistimo $\wedge$ , operacije nad bitima

```
a = 0b0101
```

```
a
```

```
b = 0b0011
```

```
b
```

```
a & b
```

```
bin(a & b)
```

```
bin(a | b)
```

```
bin(a ^ b)
```

```
bin(0)
```

```
bin(~0)
```

```
bin(2)
```

```
bin(~2)
```

```
~2
```

```
2 << 1
```

```
2 << 4
```

```
32 >> 2
```

```
3 >> 1
```

# Python, a sada nesto sasvim drugačije: kompleksni brojevi

```
#j*j  
1j*1j  
2J * 2J  
type(1J)  
abs(3+4j)  
complex(1,2)  
a = 2 + 3j  
type(a)  
a.real  
a.imag  
a.conjugate()  
a * a.conjugate()  
del a  
#type(a)
```

# Python, malo ozbiljnija matematika, moduli

```
sin(1)
import math
type(math)
dir(math)
help(math)
help(math.sin)
math.sin(1)
math.e
math.pi
math.sin(math.pi/2)
#math.exp(math.pi*1j)+1
math.cos(math.pi) + 1j * math.sin(math.pi) + 1
```

# Python, namespaces

```
del math
import math as m
m.sin(m.pi / 4) ** 2
m.exp(1) - m.e
del m
from math import *
sin(pi / 4) ** 2
exp(1) - e
e
e = 32
e
pi
pi = 14
pi
```

# Python, assignment operators

```
a = 1
a += 1
print(a)
a *= 2
print(a)
a /= 2
print(a)
a -= 4
print(a)
a **= 3
a %= 3
print(a)
-8 / 3
a = 11.0
a //= 3
print(a)
```



# Python, funkcije

```
def pdv(x):  
    return x * 1.20
```

```
type(pdv)  
pdv(100)  
pdv(150)
```

# Python, funkcije, help

```
def pdv(x):  
    'ovo je funkcija koja racuna pdv'  
    return x * 1.20
```

```
pdv(100)  
help(pdv)
```

# Python, funkcije, help u više redova

```
def pdv(x):  
    '''ovo je funkcija koja racuna pdv  
  
    a pdv je porez na dodatu vrednost'''  
    return x * 1.20
```

```
pdv(100)  
help(pdv)
```

# Python, funkcije, opcioni argumenti

```
def pdv(x, stopa = 20):  
    return x*(1 + stopa/100)
```

```
pdv(100)  
pdv(150)
```

```
def pdv(x, stopa = 20):  
    return x * (1 + stopa / 100.)
```

```
pdv(100)  
pdv(150)  
pdv(100, stopa = 23)  
pdv(100, 23)
```

```
del pdv  
pdv(10)
```

# Python, kontrola toka

```
def parnost(n):  
    if n // 2 * 2 == n:  
        print('paran')  
    else:  
        print('neparan')
```

```
parnost(4)
```

```
parnost(5)
```

```
parnost(4.2)
```

```
parnost(5.1)
```

# Python, ispitivanje tipa

```
def parnost(n):  
    if type(n) != "<type 'int'>":  
        print('argument nije ceo broj')  
        return  
    if n // 2 * 2 == n:  
        print('paran')  
    else:  
        print('neparan')
```

```
parnost(4.2)
```

```
parnost(4)
```

```
parnost(3)
```

```
type(4)
```

```
type(type(4))
```

```
type("<type 'int'>")
```

# Python, ispitivanje tipa, sada radi

```
def parnost(n):  
    if str(type(n)) != "<type 'int'>":  
        print('argument nije ceo broj')  
        return  
    if n // 2 * 2 == n:  
        print('paran')  
    else:  
        print('neparan')
```

```
parnost(4.2)
```

```
parnost(4)
```

```
parnost(3)
```

# Python, ispitivanje tipa, može i ovako

```
def parnost(n):  
    if type(n) != type(1):  
        print('argument nije ceo broj')  
        return  
    if n // 2 * 2 == n:  
        print('paran')  
    else:  
        print('neparan')
```

```
parnost(4.2)
```

```
parnost(4)
```

```
parnost(3)
```

```
parnost(4.)
```



# Python, konverzije tipova i još ponešto

```
del parnost
int(-4.2)
int(4.2)
#long(_)
float(_)
float(5)
divmod(10, 3)
divmod(12, 3)
pow(2, 8)
2**8
str(float(2**8))
```

# Python, liste

```
a = [1, 2, 5, 6]
type(a)
a[0]
a[1]
a[2]
a[3]
a[4]
a[-1]
a[-2]
a[-3]
a[-4]
a[-5]
print(a)
len(a)
```

# Python, liste, slicing and mutability

```
a[1:3]
a[1 : 2]
a[1 : - 2]
a[2 : ]
a[:2]
a[:-2]
a[3] = 7
print(a)
```

# Python, liste, dodavanje i brisanje elemenata

```
a + 9
a + [9]
a = a + [9]
len(a)
del a[(len(a) - 1)]
print(a)
del a[1]
print(a)
len(a)
```

# Python, liste, metodi append i extend

```
a = [1, 2, 3, 4]
a.append(5)
print(a)
b = [6, 7]
a.append(b)
print(a)
len(a)
del a[5]
a.extend(b)
print(a)
len(a)
del a[5:]
print(a)
```

# Python, liste, range

```
a = range(5)
len(a)
print(a)
type(a)
a = list(range(5))
len(a)
print(a)
type(a)
a = tuple(range(5))
len(a)
print(a)
type(a)
a = range(4, 10)
len(a)
print(a)
a = range(3, 10, 2)
print(a)
a = range(10, 0, -2)
print(a)
```

# Python, stack

```
a = []  
type(a)  
a.append(1)  
a.append(2)  
a.append(3)  
a.pop()  
a.pop()  
print(a)  
a = list(range(10))  
a.pop(3)  
print(a)
```

# Python, liste, insert

```
a = list(range(10))
a.insert(3, 4)
print(a)
a.insert(0, 1)
print(a)
a.insert(len(a), 'kraj')
print(a)
```



# Python, liste, reverse, sort

```
a = list(range(10))  
a.reverse()  
print(a)  
a.reverse()  
print(a)  
a = [3, 4, 2, 1]  
a.sort()  
print(a)
```

# Python, liste, brojanje i brisanje

```
a = [3, 2, 3, 1, 4, 3, 2, 2, 5, 2]
a.count(2)
a.count(3)
a.remove(3)
a.count(3)
print(a)
a.remove(3)
print(a)
a.remove(3)
print(a)
#a.remove(3)
```

# Python, in operator

```
3 in a
```

```
4 in a
```

```
a.remove(4)
```

```
4 in a
```

# Python, liste, index metod

```
print(a)  
a.index(2)  
a.index(5)  
a.index(1)  
a.index(3)
```

# Python, aliases

```
a = 3
b = a
a is b
a == b
id(a)
id(b)
help(id)
b += 1
a == b
a is b
id(a)
id(b)
```

# Python, aliases with lists

```
a = [1, 2, 3]
b = a
a is b
a == b
b[1] = 0
a == b
print(a)
a is b
c = a[:]
c == a
c is a
c[1] = 2
c == a
print(c)
print(a)
```

# Python, matrice

```
a = [[1, 2], [3, 4]]  
len(a)  
len(a[1])  
print(a[1][1])  
print(a[0][0])  
a[0, 0]
```

# Python, inicijalizacija nizova

```
a = []  
print(a)  
a = [0] * 10  
print(a)  
a = [[1] * 3] * 3  
print(a)
```



# Python, for petlja

```
a = range(10)
for i in a:
    print(i + 1, '/', len(a))
```

ili

```
a = list(range(10))
for i in a:
    print(i + 1, '/', len(a))
```

# Python, for petlja, over string, “iterable”

```
a = 'neobicno bas'  
for znak in a:  
    print(znak)
```

# Python, if-else

```
a = 'abrakadabra'
b = ''
for znak in a:
    if znak != 'a':
        b += znak
    else:
        b += '_'
print(b)
```

# Python, if-elif-else

```
a = 'abrakadabra'
b = ''
for znak in a:
    if znak == 'a':
        b += '_'
    elif znak == 'k':
        b += '*'
    else:
        b += znak
print(b)
```